10-2020

Camera Placement Meeting Restrictions of Computer Vision

Sara Aghajanzadeh
Purdue University

Roopasree Naidu
Purdue University

Shuo-Han Chen
Academica Sinica, Taipei, Taiwan

Caleb Tung
Purdue University

Abhinav Goel
Purdue University

See next page for additional authors

Recommended Citation

Sara Aghajanzadeh, Roopasree Naidu, Shuo-Han Chen, Caleb Tung, Abhinav Goel, Yung-Hsiang Lu, George K. Thiruvathukal, Camera Placement Meeting Restrictions of Computer Vision, Proceedings of IEEE International Conference on Image Processing 2020.

CAMERA PLACEMENT MEETING RESTRICTIONS OF COMPUTER VISION

Sara Aghajanzadeh, Roopasree Naidu, Shuo-Han Chen[1]*, Caleb Tung,*
Abhinav Goel, Yung-Hsiang Lu, George K. Thiruvathukal[2]

School of Electrical and Computer Engineering, Purdue University, West Lafayette, USA
[1]Institute of Information Science, Academia Sinica, Taipei, Taiwan
[2]Department of Computer Science, Loyola University Chicago, Chicago, USA

ABSTRACT

In the blooming era of smart edge devices, surveillance cameras have been deployed in many locations. Surveillance cameras are most useful when they are spaced out to maximize coverage of an area. However, deciding where to place cameras is an NP-hard problem and researchers have proposed heuristic solutions. Existing work does not consider a significant restriction of computer vision: in order to track a moving object, the object must occupy enough pixels. The number of pixels depends on many factors (how far away is the object? What is the camera resolution? What is the focal length?). In this study we propose a camera placement method that not only identifies effective camera placement in arbitrary spaces, but can account for different camera types as well. Our strategy represents spaces as polygons, then uses a greedy algorithm to partition the polygons and determine the cameras' locations to provide desired coverage. The solution also makes it possible to perform object tracking via overlapping camera placement. Our method is evaluated against complex shapes and real-world museum floor plans, achieving up to 82% coverage and 28% overlap.

Index Terms— Computational Geometry, Art Gallery Problem, Computer Vision, Camera Placement, Multimedia Surveillance System

1. INTRODUCTION

Smart edge devices equipped with cameras are utilized in automated surveillance systems to gather visual data and process the data with computer vision techniques, such as object detection or tracking. Erdem et al. [1] note that placing surveillance cameras at the proper locations is important for effective object detection and tracking. Existing work on camera placement assumes that a surveillance camera can see infinitely far away (similar to the original "art gallery problem"). Realistically, computer vision is ineffective when objects are too far away and too small [2]. Even human eyes end up having to squint to see things far away! Most tracking applications struggle to detect objects at low resolutions (below 10 pixels per foot) [3]. An example is given in Fig. 1. In Fig. 1 (a), the object in the bounding box has a resolution of 20 pixels per foot, thus the object of interest can be tracked easily. In Fig. 1 (b), the object is too far away from the camera and has a resolution below 10 pixels per foot, making it difficult to track effectively. Generally speaking, when the number of pixels of a bounding box is less than 400, the resolution is considered low [4]. More importantly, visibility is not sufficient for automated persistent tracking. This restriction is further complicated by the fact that surveillance cameras come in all types: differing focal lengths and camera resolutions all have an impact.

(a) (b)

Fig. 1: Comparison of the number of pixels per object used by an object tracker.

The camera placement problem has been proven to be an NP-hard problem; thus, instead of seeking an optimal solution, the placement techniques seek an approximate near-optimal solution [5]. Most of the previous approaches are either limited to the trade-offs between coverage and costs or the prior knowledge given by security experts [6]. This paper proposes a fast algorithm to determine where to place surveillance cameras to achieve good coverage of a space. This novel algorithm accounts for the restrictions of computer vision by considering the *effective* field of coverage (FOC) of a camera based on the camera's specifications and the effective range (distance from camera) required to successfully identify and track objects. This algorithm accepts a polygon representation of the space and divides the area into smaller polygons of the same size. A greedy strategy is utilized to find the camera locations that can satisfy the requirements for each subpolygon. Each subpolygon has at least one camera for surveillance. Since each subpolygon may have an arbitrary shape and the camera's field of coverage is a triangle,

the 100% coverage may not be achieved. The effectiveness of the proposed greedy solution is evaluated through experiments on the real-world floor plan of the Louvre Abu Dhabi museum. The experiments show that the proposed solution has consistent results, always between 67% to 82% coverage and 18% to 28% overlap for any n-sided polygon.

2. BACKGROUND AND CONTRIBUTIONS

The problem of automated camera placement has captured the attention of the research community for quite some time. Bisagno et al. [7] used reconfigurable cameras that can dynamically adapt their field of view (FOV), resolution, and position to provide coverage by focusing attention on critical areas of a crowd while ensuring an acceptable level of attention on less critical areas, resulting in a trade-off between coverage and resolution. Yabuta et al. [5] proposed a method considering camera specifications and a trade-off between coverage and the cost (i.e., the number of cameras). And Altahir et al. [8] propsed a dynamic programming solution that relies on human experts to determine camera locations.

To the authors' knowledge, no existing solution has been developed that can provide an optimal solution (i.e., using the least number of cameras) to *fully* cover polygons of arbitrary shapes. Although a significant amount of research has been conducted in the theoretical aspects of the problem, few studies have been devoted to automated placement with the consideration of vision technologies. In fact, automation could be useful in cases of frequent change in deployment plans. The majority of theoretical solutions assume unlimited field of coverage and infinite visibility; thus, existing theoretical works cannot be applied in a real deployment [1]. This paper proposes a greedy solution for determining an effective placement of cameras for monitoring an area with a target resolution sufficient for computerized tracking of individuals.

3. GREEDY CAMERA PLACEMENT

As stated previously [9], the solution for 100% coverage is computationally intractable, so this paper does not produce full coverage. Instead, this paper aims to achieve a balance between coverage and resolution. Fig. 2 presents a flowchart of the proposed approach. The input to the algorithm are the desired floor plan as a mesh of 2-dimensional grid points, producing a mapping between them to create the original polygon, and the camera specifications. The main computational steps are: 1) Initialization procedure - computing camera FOC based on the camera model, and splitting the polygon into equal areas based on the FOC; and 2) Placement procedure - placing cameras using a greedy strategy in each subpolygon. The output of the program provides the total number of cameras, their locations, % covered, and % overlapped.

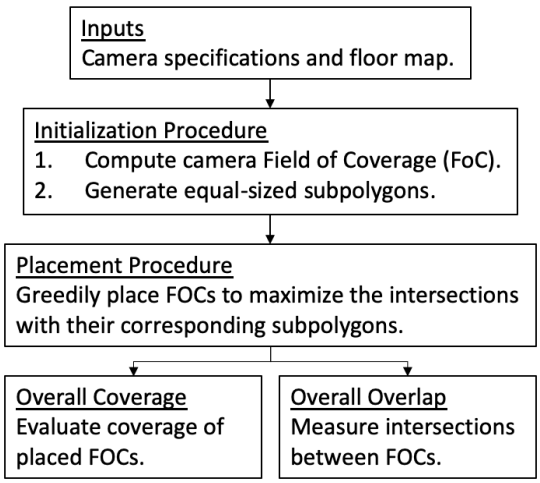


Fig. 2: Flow chart of the proposed approach.

3.1. Camera Placement Procedure

this distance (the camera specifications are depicted in Fig. 3). Based on the observation from Section 1, this paper sets a threshold of 20 pixels per foot. Given the threshold, we determine the distance from a camera to a target individual that is required for successful detection and tracking. We use Equation 1 to compute this distance. Fig 3 shows the relationship between the camera specifications. Given the distance, we compute the FOC, which is the area of a triangle calculated using $Area = \frac{1}{2} \times b \times h$, where h is the camera distance and b is the maximum horizontal field of view calculated using $\frac{\text{Horizontal camera resolution}}{\text{PPF}} = \text{FOV}$ for each camera.

$$\text{focal length} = \frac{\text{distance} \times \text{chip-width} \times \text{resolution}}{\text{number-pixels}_{\,horizontal}} \quad (1)$$

After converting a floor plan into a 2-dimensional n-sided polygon, it is theoretically possible to consider all grid points as possible camera positions; however, it is not practical or efficient due to the increased computational overhead. Our novel approach involves splitting the space into equal regions based on the camera FOC, which is the coverage (meeting restrictions of computer vision) provided by a single camera and is derived in advance from the cameras' specifications. Thus, each divided subpolygon is created with the same area.

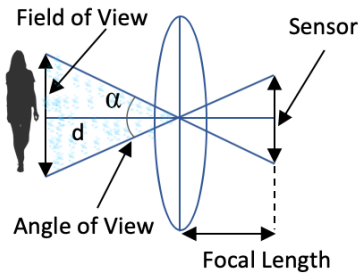


Fig. 3: The input camera specifications.

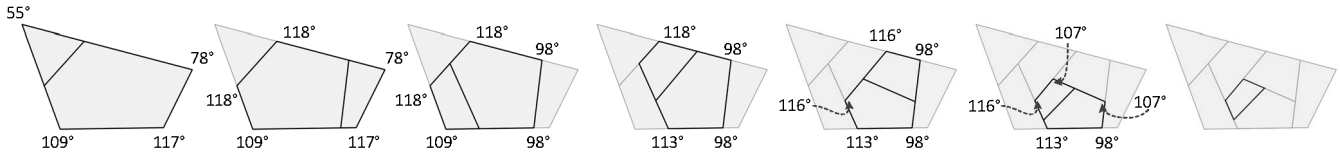


Fig. 4: Steps showing the original polygon (far left) being split into subpolygons with equal areas. The area is based on the calculated FOC subject to camera viewing angle α and distance d required to maintain successful tracking.

We use Equation 2 to determine the area of the main polygon.

$$\text{Area}_{\text{polygon}} = \left| \frac{(x_1y_2 - y_1x_2) + ... + (x_ny_1 - y_nx_n)}{2} \right| \quad (2)$$

where x_i and y_i are the coordinates of a vertex. Next, we use the algorithm in [10], offering a closed-form solution to splitting a polygon into any number of equal areas, to divide the main polygon into n subpolygons with areas equal to $\frac{\text{Area}_{mainPolygon}}{n}$. The process is displayed in Fig. 4.

For every vertex of the subpolygons, the level of intersection between the camera's FOC and the subpolygon is computed, and the FOC is rotated such that it lies inside the polygon and provides the best intersection between the FOC and its corresponding subpolygon. Fig. 5 summarizes the various possibilities for the rotated FOC.

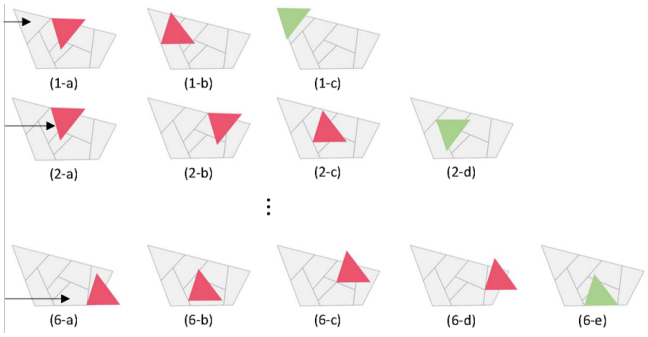


Fig. 5: For every subpolygon, indicated by the arrow, the solution examines each vertex [a-e] and computes the coverage gained by placing camera FOC at that vertex. The vertex with the highest coverage score is chosen for each subpolygon. Red color designates ineffective placements whereas green color designates the effective placement. As an example, (1-a) camera FOC does not intersect with its own subpolygon at all, thus the solution proceeds to the next vertex (1-b), where camera FOC intersects with the subpolygon over a small area. The solution proceeds to the next vertex (1-c), where highest intersection is achieved. So, it picks this vertex. The same procedure continues for all the other subpolygons until each subpolygon has at least one FOC placed at one of its vertices.

4. EVALUATION

The proposed greedy solution provides a functional balance between coverage and resolution requirement. The greedy approach offers relatively high efficiency and exhibits a low runtime. The computational complexity is O(nm), n being the number of subpolygons and m being the number of their vertices. It seeks the candidates that satisfy the objective locally rather than identifying a global optimum. For our evaluation, a 1080p HD fixed camera with angle of view $\alpha = 74°$and focal length $f = 3.6$mm, distance $d = 72$ ft limited for resolution requirement is used. Thus, the camera's FOC covers an area of 3,456 square feet. This section evaluates the proposed method in two different scenarios.

4.1. Base Case Evaluation

Fig. 6 shows an evaluation of coverage and overlap achieved by the greedy placement. Fig. 6 (a) shows the ideal solution with 100% coverage without overlap, plotted by humans compared with our solution, Fig. 6 (b). Both solutions use the same number of cameras, since the original polygon is divided based on the camera FOC. Our method provides less than 100% coverage because the subpolygons created by the algorithm do not always match triangle shapes of the camera FOC; as a result, the proposed method has overlaps, which could be potentially useful for tracking applications during camera hand-off. Table 1 shows the exact amount of coverage

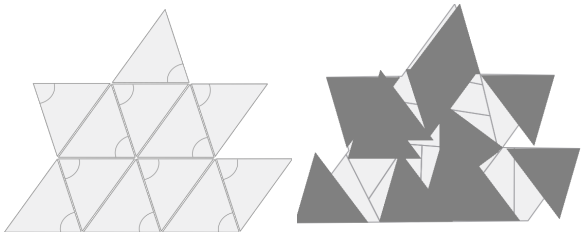


Fig. 6: (a) The ideal solution for the polygon to achieve 100 % coverage. (b) Our camera placement method uses the same number of cameras (12) and achieves 72% coverage with 25% overlap.

and overlap with increasing number of cameras. Furthermore, Fig. 7 shows the performance of the algorithm as the number of edges increase, resulting in more complex polygons and requiring more cameras. Our solution stabilizes at approximately 70% coverage with 30% overlap when compared with the perfect solution.

Layout	Number of Cameras		Coverage	Overlap
	Best Solution	Our Solution		
-	6	6	82.00%	17.60%
Fig. 6	12	12	71.95%	25.23%
-	18	18	67.96%	27.94%

Table 1: Camera coverage and overlap outcome.

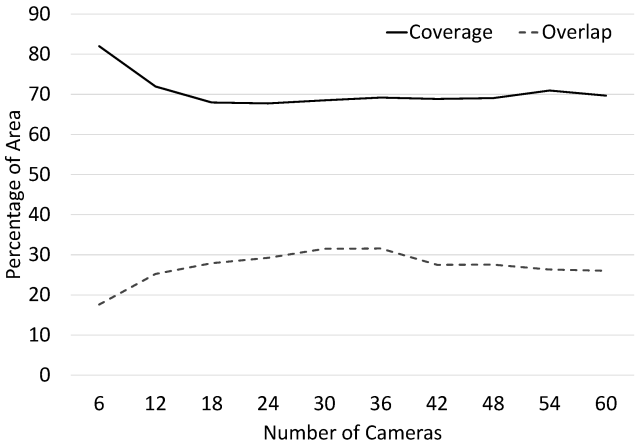


Fig. 7: Evaluation of coverage and overlap with increasing number of cameras as a result of more complex polygon (more edges and vertices)

4.2. Complex Case Evaluation

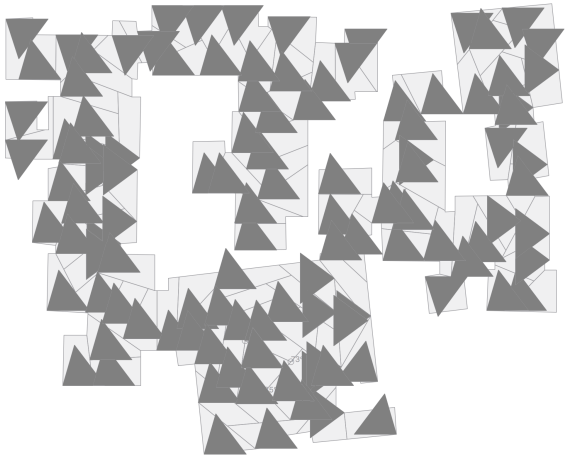
In this section, we present our approach applied to the floor plan of the Louvre museum shown in Fig. 8 (a). Fig. 8 (b) shows the result of dividing the floor plan into equal subpolygons based on the camera specifications, and Fig. 8 (c) illustrates the placement of camera FOCs such that effective coverage is achieved and requirements of computer vision applications are met.

5. CONCLUSION

Compared with prior works on the surveillance camera placement problem, our automated placement method takes into consideration the realistic constraints of computer vision, making our algorithm suitable for real-world deployment. Our solution eliminates the gap between theoretical computational geometry and the realistic requirements of computer vision by ensuring both the minimum required resolution and the camera angle of view coverage are satisfied during camera placement. To achieve above goals, the proposed greedy solution partitions the main polygon into fixed size subpolygons and then cameras are greedily placed within the subpolygons. The proposed solution is implemented and evaluated on a real-world floor plan. The evaluation results show that the greedy solution can achieve 67% to 82% coverage and 18% to 28% overlap for spaces of different shapes.

(a) Floor plan of Louvre Abu Dhabi museum (source: https://www.archdaily.com).

(b) Original polygon is split into subpolygons with equal areas. Area is based on the camera FOC subject to camera viewing angle α and distance d required to maintain successful detection and tracking.



(c) Greedy placement of the cameras (viewing angle α = 74°and distance $d = 72\,ft$) for each subpolygon. Dark gray areas are covered by the cameras. 66.74% of the area is covered and 21.48% overlap is achieved.

Fig. 8: Complex Case Evaluation

6. REFERENCES

[1] U. M. Erdem and S. Sclaroff, "Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements," *Computer Vision and Image Understanding*, vol. 103, no. 3, pp. 156 – 169, 2006, special issue on Omnidirectional Vision and Camera Networks.

[2] R. Cucchiara, "Multimedia surveillance systems," in *Proceedings of the Third ACM International Workshop on Video Surveillance and Sensor Networks*, 2005.

[3] N. Jiang, W. Liu, H. Su, and Y. Wu, "Tracking low resolution objects by metric preservation," in *CVPR 2011*, June 2011, pp. 1329–1336.

[4] Y. Wu, J. Lim, and M. Yang, "Online object tracking: A benchmark," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2411–2418.

[5] K. Yabuta and H. Kitazawaand, "Optimum camera placement considering camera specification for security monitoring," in *2008 IEEE International Symposium on Circuits and Systems*, May 2008, pp. 2114–2117.

[6] K. Tarabanis and R. Y. Tsai, "Computing viewpoints that satisfy optical constraints," in *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1991, pp. 152–158.

[7] N. Bisagno, N. Conci, and B. Rinner, "Dynamic camera network reconfiguration for crowd surveillance," in *Proceedings of the 12th International Conference on Distributed Smart Cameras*, 2018, pp. 4:1–4:6.

[8] A. A. Altahir, V. S. Asirvadam, N. H. B. Hamid, P. Sebastian, N. B. Saad, R. B. Ibrahim, and S. C. Dass, "Optimizing visual surveillance sensor coverage using dynamic programming," *IEEE Sensors Journal*, vol. 17, no. 11, pp. 3398–3405, 2017.

[9] S. Eidenbenz, C. Stamm, and P. Widmayer, "Inapproximability of some art gallery problems," in *Proceedings of the 10th Canadian Conference of Computational Geometry*, 1998, pp. 64–65.

[10] S. Khetarpal, "Dividing a polygon in any given number of equal areas," 2014. [Online]. Available: http://www.khetarpal.org/polygon-splitting/